UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/353,535	07/14/1999	JOSHUA J. BLOCH	SUN1P247/P41	3724

22434 7590 08/15/2003

BEYER WEAVER & THOMAS LLP
P.O. BOX 778
BERKELEY, CA 94704-0778

EXAMINER
ALI, SYED J

ART UNIT	PAPER NUMBER
2127	

DATE MAILED: 08/15/2003

4

Please find below and/or attached an Office communication concerning this application or proceeding.

3

Office Action Summary	Application No.	Applicant(s)	
	09/353,535	BLOCH, JOSHUA J.	
	Examiner	Art Unit	
	Syed J Ali	2127	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 July 1999* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-33* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-7,10-14,17-24 and 27-33* is/are rejected.

7)☒ Claim(s) *8,9,15,16,25 and 26* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☒ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 2 .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

Claim Rejections - 35 USC § 112

1. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

2. Claims 1, 18, and 28 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

As per claim 1, the exact meaning of the limitation "for each thread-local variable, mapping each thread to a value" is unclear. The first portion of the limitation would suggest that the mapping action is related to each thread-local variable, yet the second portion of the limitation seems to indicate that the entire thread is mapped to a variable. For purposes of examining this claim, it will be interpreted that the intended meaning of this limitation was meant to be "for each thread-local variable, mapping each thread-local variable to a value".

As per claims 18 and 28, similar limitations are claimed as those discussed above in claim 1. They are indefinite for the same reasons discussed above in reference to claim 1.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

4. Claims 1-2, 6, 12-13, 22-23, and 33 are rejected under 35 U.S.C. 102(a) as being anticipated by Torii (USPN 5,913,059).

As per claim 1, Torii discloses a method for providing inheritable thread-local storage from a parent thread to a child thread, the method comprising:

for each thread-local variable, mapping each thread to a value (col. 5 lines 57-63, "when the content of the register file is inherited at the time of fork, all of the contents of the register file at the time of fork are physically inherited to the register file in the processor which executes a newly generated thread"); and

when a parent thread creates a child thread, automatically iterating over the parent thread's values to create the child thread's initial values (col. 6 line 54 - col. 8 line 2, "In the cycle 3, before the writing to the register r1 is performed in the processor #0 [9a] which is executing the parent thread, reading is performed to transfer the content of the register r1 to the processor #1 [9a] [see [D] in the first half of the cycle 3]. The content of the register r1 is transferred to the register file #1 [13b] of the processor #1 [9b], and is written thereto", wherein this process continues through all the registers of the parent thread until all the register values are inherited to the child thread).

As per claim 2, Torii discloses the method of Claim 1, wherein the step of mapping comprises maintaining a map, associated with each thread object, that maps each thread-local variable to a value (col. 4 line 63 - col. 5 line 4, "When a thread 1 [a parent thread] generating a new thread 3 [a child thread] executes a thread generation instruction 2 [a 'fork' instruction] on a course of an execution flow to generate the new thread 3, the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread"); and the step of iterating comprises iterating over the map (as discussed above in reference to claim 1, the mapping operation copies the contents of the register iteratively).

As per claim 6, Torii discloses the method of Claim 1, wherein a child thread's initial value is a copy of a corresponding parent thread's value (col. 4 line 63 - col. 5 line 4, "the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread").

As per claim 12, Torii discloses a method for providing automatic value inheritance when a parent thread creates a child thread, comprising:

associating, for each thread object, each thread-local variable with a value (col. 5 lines 57-63, "when the content of the register file is inherited at the time of fork, all of the contents of the register file at the time of fork are physically inherited to the register file in the processor which executes a newly generated thread"); and

automatically iterating over the thread-local values to create a child value (col. 6 line 54 - col. 8 line 2, "In the cycle 3, before the writing to the register r1 is performed in

the processor #0 [9a] which is executing the parent thread, reading is performed to transfer the content of the register r1 to the processor #1 [9a] [see [D] in the first half of the cycle 3]. The content of the register r1 is transferred to the register file #1 [13b] of the processor #1 [9b], and is written thereto").

As per claim 13, Torii discloses the method of Claim 12, wherein the child value is a copy of the corresponding parent value (col. 4 line 63 - col. 5 line 4, "the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread").

As per claims 22-23, a computer readable medium including program code for implementing the method of claims 12-13 is disclosed in Torii. Specifically, Torii is disclosed in a multi-processor system implementing multithreading. It is therefore inherent in the disclosure of Torii that a computer readable medium must exist in order to make the method disclosed by Torii functional.

As per claim 33, Torii discloses a method for providing automatic inheritance of parent thread-local values to a child thread upon child thread creation, wherein a parent thread is associated with the parent's thread-local values, the method comprising:

determining the parent's inheritable thread-local values (col. 5 lines 57-63, "when the content of the register file is inherited at the time of fork, all of the contents of the register file at the time of fork are physically inherited to the register file in the processor

which executes a newly generated thread", wherein the contents of the register file associated with the parent thread are the values that are inheritable); and

automatically initializing the child's thread-local values corresponding to the parent's inheritable thread-local values, upon child creation, based on a predetermined child value method (col. 6 line 54 - col. 8 line 2, "In the cycle 3, before the writing to the register r1 is performed in the processor #0 [9a] which is executing the parent thread, reading is performed to transfer the content of the register r1 to the processor #1 [9a] [see [D] in the first half of the cycle 3]. The content of the register r1 is transferred to the register file #1 [13b] of the processor #1 [9b], and is written thereto").

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 3, 18-19, and 28-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Torii in view of Kim et al. (USPN 6,553,531) (hereinafter Kim).

As per claim 3, Torii discloses the method of Claim 1, wherein the step of mapping comprises maintaining a map, associated with each thread-local variable, that maps each thread to a value (col. 4 line 63 - col. 5 line 4, "When a thread 1 [a parent thread] generating a new thread 3 [a child thread] executes a thread generation instruction

2 [a 'fork' instruction] on a course of an execution flow to generate the new thread 3, the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread").

Kim discloses the following limitations not shown by Torii, specifically wherein for each thread a linked list is maintained, the linked list linking inheritable thread-local values associated with the thread (col. 13 line 21 - col. 14 line 46, "A chain of records, each called a 'HLevel_record' [standing for 'hierarchy level record'], are created with one such record being created for each level of the class hierarchy.", "The HLevel_record's are connected as a doubly linked list"); and wherein the step of iterating comprises iterating over the linked list (wherein this limitation would be performed in conjunction with what is disclosed in Torii, and is addressed further below).

It would have been obvious to one of ordinary skill in the art to combine Torii with Kim since by storing the variables associated with each thread in a doubly linked list of records, corresponding to different levels of hierarchy as disclosed by Kim (col. 13 lines 47-64) allows both a simple traversal of the variables corresponding to each thread by simply following the links as well as providing a way to easily trace inheritance since the doubly linked list also allows traversal of the records in a hierarchical fashion. Although Torii is related specifically to the inheriting of data via processor register files, it would be a simple modification to have the data of each register defined in a linked list.

As per claim 18, Torii discloses a method for providing automatic value inheritance when a parent thread creates a child thread, the method comprising:

associating, for each thread-local variable, each thread to a value (col. 4 line 63 - col. 5 line 4, "When a thread 1 [a parent thread] generating a new thread 3 [a child thread] executes a thread generation instruction 2 [a 'fork' instruction] on a course of an execution flow to generate the new thread 3, the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread").

Kim discloses the following limitations not shown by Torii, specifically wherein for each thread a linked list is maintained, the linked list linking inheritable thread-local values associated with the thread (col. 13 line 21 - col. 14 line 46, "A chain of records, each called a 'HLevel_record' [standing for 'hierarchy level record'], are created with one such record being created for each level of the class hierarchy.", "The HLevel_record's are connected as a doubly linked list"); and

automatically iterating over the linked list to create a child value corresponding to each inheritable parent value when a child is created (performed in conjunction with the disclosure of Torii, as discussed above in reference to claim 3).

Furthermore, the motivation for combining Torii and Kim is provided above in reference to claim 3.

As per claim 19, Torii discloses the method of Claim 18, wherein the child value is a copy of the corresponding parent value (col. 4 line 63 - col. 5 line 4, "the content at the time when the parent thread 1 executes the fork instruction 2 is inherited to a register file of the child thread").

As per claims 28-29, a computer readable medium including program code for implementing the method of claims 18-19 is disclosed in Torii. Specifically, Torii is disclosed in a multi-processor system implementing multithreading. It is therefore inherent in the disclosure of Torii that a computer readable medium must exist in order to make the method disclosed by Torii functional.

7. Claims 4-5, 10-11, 17, 21, 27, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Torii in view of Kim in view of De Pauw.

As per claim 4, De Pauw discloses the following limitations not shown by Torii, specifically the method of Claim 2, wherein the map comprises a hash table (col. 11 lines 7-31, "Each linked list represents a hash bucket, and is composed of HashtableEntry objects", wherein the hash table object is used to quickly reference data, and could be used in conjunction with the disclosure of Kim to effectively speed up the inheritance process, as discussed below).

It would have been obvious to one of ordinary skill in the art to combine the modified Torii and De Pauw since by modifying the doubly linked list of Kim, as discussed in reference to claim 3, a hash table object reflective of the relationship of those linked lists would significantly improve the performance of the inheritance procedure. That is, when a thread creates a child thread, there should be some mechanism in place to locate the data of the parent thread for initializing the values of the child thread. In the case of Torii, this is done through a register file, and iterated through one register at a time. As discussed in reference to claim 3, this register file could be modified to define

the data of the register file as a linked list. Nonetheless, the problem remains that the data has to be iterated through sequentially. In the case where there are many threads of execution, the search time to locate the thread of interest could be unreasonably high. By further modifying the linked list to be a hash table wherein the hash table is an array of linked lists, a unique identifier could identify each thread, and a hashing function could tell the inheritance procedure exactly where to find the data that is to be inherited. Furthermore, since the hash table is organized as a linked list, the pertinent data is then connected to the head of the linked list and all data that is to be inherited can be found in minimal time. Please reference Fig. 16 of De Pauw for an indication of how a hash table can be used to indicate depth, similar to that of inheritance and parent/child pairs. While the claim limitations of the parent claim 2 do not specifically refer to linked lists, the disclosure of Kim is pertinent to the present claim, as it shows how a linked list can be used to implement inheritance. To that end, the combination of Torii, Kim, and De Pauw show how a hash table, implemented as an array of linked lists, can simplify inheritance.

The motivation for combining Torii, Kim, and De Pauw discussed above also applies to the remainder of the claims rejected as unpatentable over Torii in view of Kim in view of De Pauw.

As per claim 5, De Pauw discloses the method of Claim 3, wherein the map comprises a hash table (col. 11 lines 7-31, "Each linked list represents a hash bucket, and is composed of HashtableEntry objects", wherein the hash table object is used to quickly reference data, and could be used in conjunction with the disclosure of Kim to effectively speed up the inheritance process).

As per claim 10, De Pauw discloses the method of Claim 2, wherein the method is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

As per claim 11, De Pauw discloses the method of Claim 3, wherein the method is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

As per claim 17, De Pauw discloses the method of Claim 12 wherein the method is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

As per claim 21, De Pauw discloses the method of Claim 18, wherein the method is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

As per claim 27, De Pauw discloses the method of Claim 22, wherein the computer program code is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

As per claim 31, De Pauw discloses the method of Claim 28, wherein the computer program code is implemented in a Java programming language as a class (Abstract, "Methods are provided for extracting reference patterns in JAVA and depicting the same").

8. Claims 7, 14, 24, and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Torii in view of Galloway et al. (USPN 6,378,004) (hereinafter Galloway).

As per claim 7, Galloway discloses the following limitations not shown by Torii, specifically the method of Claim 1, wherein a child thread's value is a predetermined function of a corresponding parent thread's value (col. 6 line 44 - col. 7 line 2, "The lpParameter parameter specifies a single 32-bit parameter value for an argument for the new thread", wherein the argument is applied to a function to determine the values for the child thread).

It would have been obvious to one of ordinary skill in the art to combine Torii with Galloway since under certain circumstances, the child thread that is inheriting values from a parent thread may not be intended to function in exactly the same manner as the parent thread. In that case, the child thread may rather be intended to function as some sort of derivative of the parent thread, wherein the initial values correspond to how the child thread should behave. Programming languages have functionality in place to support such a concept, such as virtual functions that override the methods of the parent class. Passing parameters to those virtual functions, as suggested by the inheritance

method disclosed by Galloway, would allow the child thread to function according to a variety of specifications, thus increasing the scalability and possible uses for a class.

The motivation for combining Torii and Galloway discussed above also applies to the remainder of the claims rejected as unpatentable over Torii in view of Galloway.

As per claim 14, Galloway discloses the following limitations not shown by Torii, specifically the method of Claim 12, wherein the child value is a function of the corresponding parent value (col. 6 line 44 - col. 7 line 2, "The lpParameter parameter specifies a single 32-bit parameter value for an argument for the new thread", wherein the argument is applied to a function to determine the values for the child thread).

As per claim 24, a computer readable medium including program code for implementing the method of claim 14 is disclosed in Torii. Specifically, Torii is disclosed in a multi-processor system implementing multithreading. It is therefore inherent in the disclosure of Torii that a computer readable medium must exist in order to make the method disclosed by Torii functional.

As per claim 32, Torii discloses a computer system providing automatic inheritance of thread-local values of a parent thread to a child thread, the computer system comprising:

a processor (Fig. 2, elements 5a and 5b); and

a computer program operating on the processor that creates child thread-local values for use by a child thread (col. 6 line 54 - col. 8 line 2, "In the cycle 3, before the

writing to the register r1 is performed in the processor #0 [9a] which is executing the parent thread, reading is performed to transfer the content of the register r1 to the processor #1 [9a] [see [D] in the first half of the cycle 3]. The content of the register r1 is transferred to the register file #1 [13b] of the processor #1 [9b], and is written thereto", wherein this process continues through all the registers of the parent thread until all the register values are inherited to the child thread).

Galloway shows the following limitations not shown by Torii, specifically, wherein the child thread-local values are a function of the parent's thread-local values (col. 6 line 44 - col. 7 line 2, "The lpParameter parameter specifies a single 32-bit parameter value for an argument for the new thread", wherein the argument is applied to a function to determine the values for the child thread).

9. Claims 20 and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Torii in view of Kim in view of Galloway.

As per claim 20, Galloway shows the following limitations not shown by the modified Torii, specifically, the method of Claim 18, wherein the child value is a function of the corresponding parent value (col. 6 line 44 - col. 7 line 2, "The lpParameter parameter specifies a single 32-bit parameter value for an argument for the new thread", wherein the argument is applied to a function to determine the values for the child thread).

It would have been obvious to one of ordinary skill in the art to add Galloway to the modified Torii since under certain circumstances, the child thread that is inheriting

values from a parent thread may not be intended to function in exactly the same manner as the parent thread. In that case, the child thread may rather be intended to function as some sort of derivative of the parent thread, wherein the initial values correspond to how the child thread should behave. Programming languages have functionality in place to support such a concept, such as virtual functions that override the methods of the parent class. Passing parameters to those virtual functions, as suggested by the inheritance method disclosed by Galloway, would allow the child thread to function according to a variety of specifications, thus increasing the scalability and possible uses for a class.

As per claim 30, a computer readable medium including program code for implementing the method of claim 20 is disclosed in Torii. Specifically, Torii is disclosed in a multi-processor system implementing multithreading. It is therefore inherent in the disclosure of Torii that a computer readable medium must exist in order to make the method disclosed by Torii functional.

Claim Objections

10. Claims 8-9, 15-16, and 25-26 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Conclusion

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Syed J Ali whose telephone number is (703) 305-8106. The examiner can normally be reached on Mon-Fri 8-5:30, 2nd Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William A Grant can be reached on (703) 308-1108. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 746-7239 for regular communications and (703) 746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Syed Ali
August 6, 2003

MAJID BANANKHAH
PRIMARY EXAMINER